Figure 1: Flow Chart

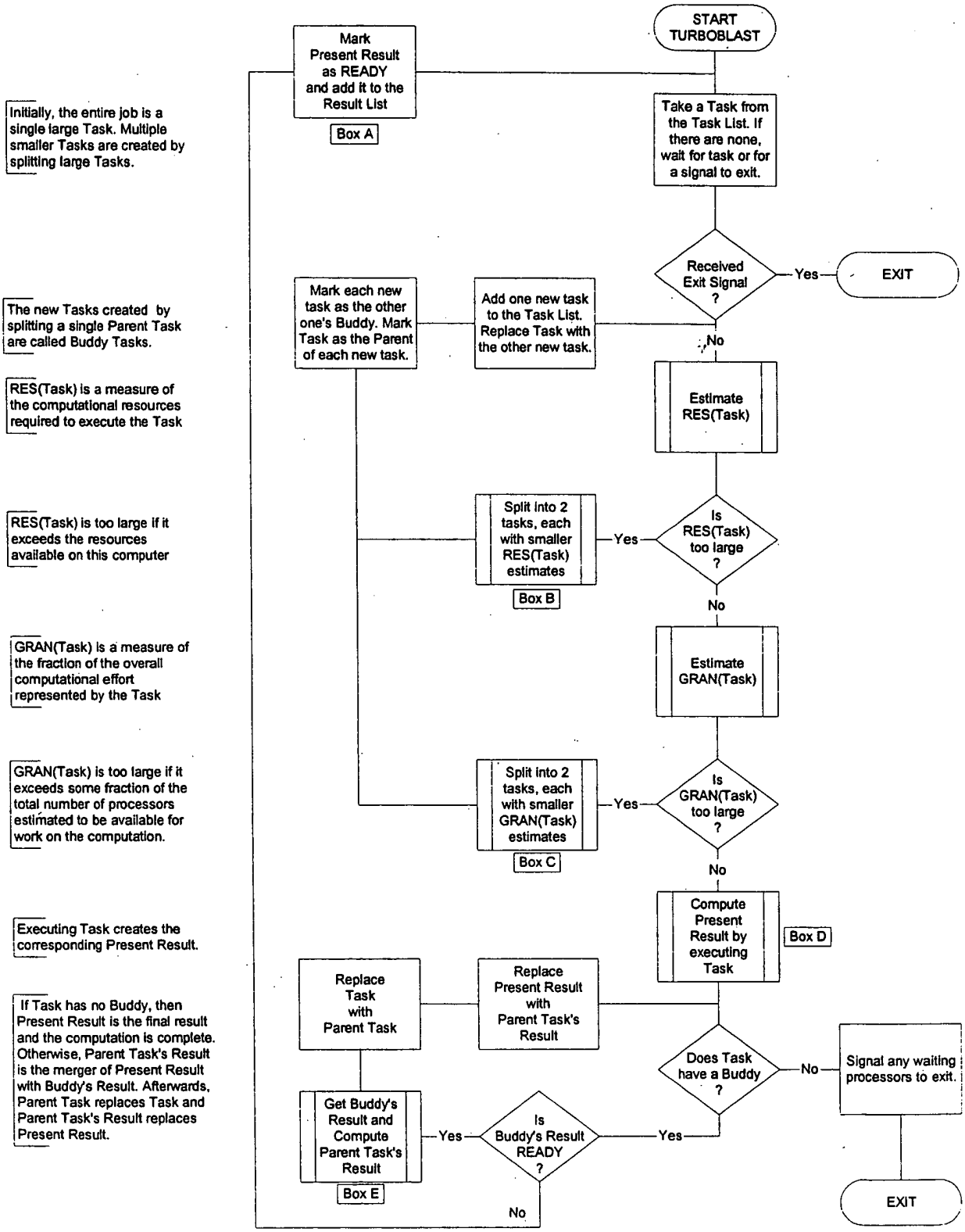
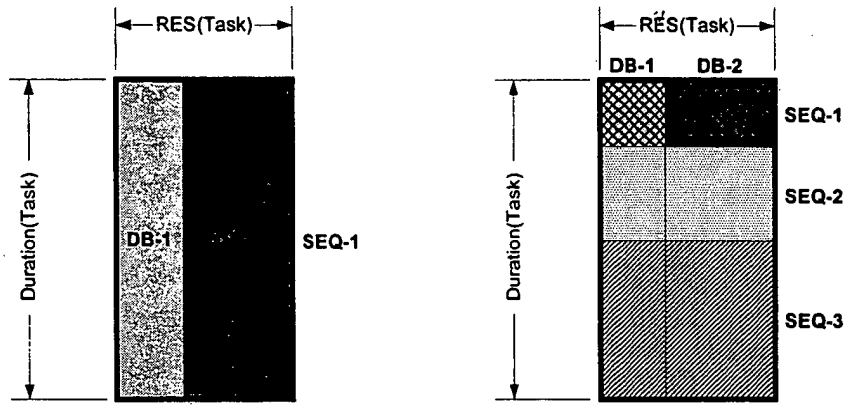


Figure 2

Rectangular Representation of Searching Tasks



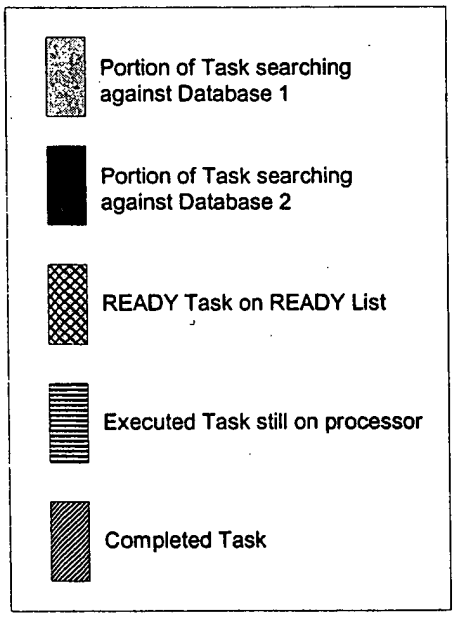
(a) Representation of one Task searching a single sequence against 2 databases

(b) Representation of one Task searching 3 sequences against 2 databases

Figure 3 (Part 1 of 7)
Detailed Illustration of Method Applied to a Searching Task

Legend

Each of the lettered subfigures (3(a) through 3(q)) shows the representation of the entire searching task at a particular time point during a sample operation of the method of the invention when run on two processors. In addition to the representation of the tasks, each part of Figure 3 also shows the contents of 2 important lists on the bulletin board (i.e., the Task List and the Result List) and indicates the current activities for each of the two participating processors at the corresponding instant of time. The processor activities are correlated with Figure 4, which illustrates the details of the processor activity and includes a time line that is correlated to the subfigures of this figure. The information below describes the markings used throughout Figure 3, and it describes the Task naming convention used in both figures.



Task Naming: The Entire Task is "Task 1".

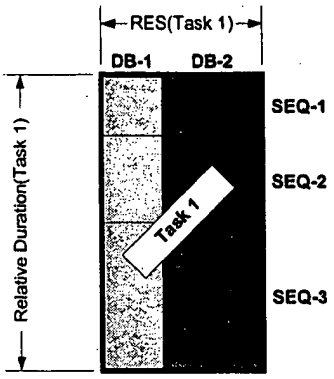
Tasks created by splitting larger divisions are denoted by names using dotted notation in which either the Parent Task's name is extended with a period (".") followed either by a capital letter or an Arabic numeral.

Capital letters are used when vertical splitting is performed based on RES(Task), as when Task 1.A and Task 1.B denote the two tasks created by subdividing Task 1.

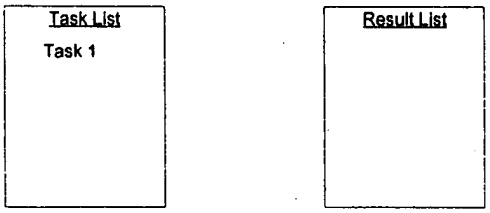
Arabic numerals are used when horizontal splitting is performed based on GRAN(Task), as when Task 1.A.1 and Task 1.A.2 denote the two tasks created by subdiving Task 1.A.

Figure 3 (Part 2 of 7)

(a)



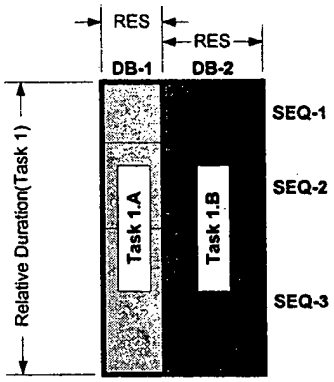
Entire Task searching 3 sequences against 2 databases



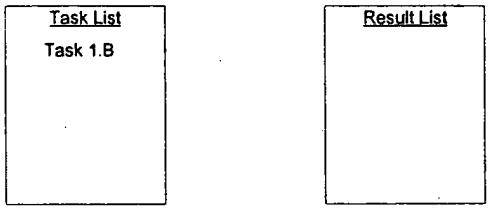
Processor 1: Currently idle

Processor 2: Currently idle

(b)



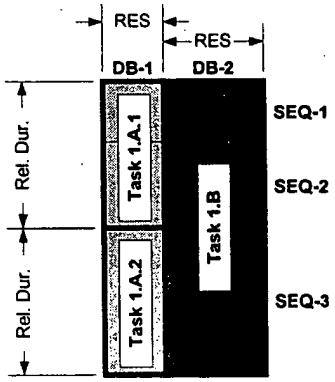
Processor 1 divides Task 1 vertically (Flowchart Box B); keeps Task 1.A



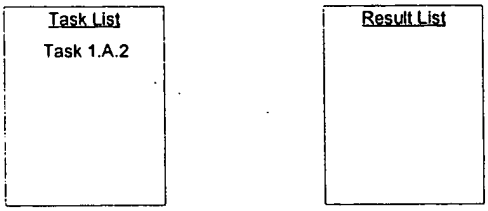
Processor 1: Done dividing Task 1.B. About to work on Task 1.A

Processor 2: Currently idle. About to work on Task 1.B

(c)



Processor 1 divides Task 1.A horizontally (Box C); k eps and begins executing Task 1.A.1

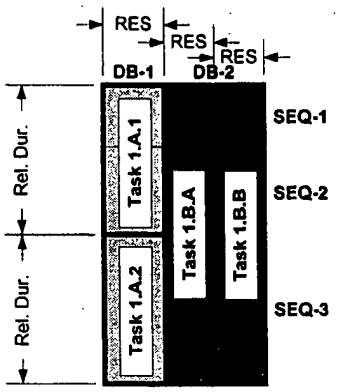


Processor 1: Done dividing Task 1.A. About to work on Task 1.A.1

Processor 2: Currently dividing Task 1.B

Figure 3 (Part 3 of 7)

(d)



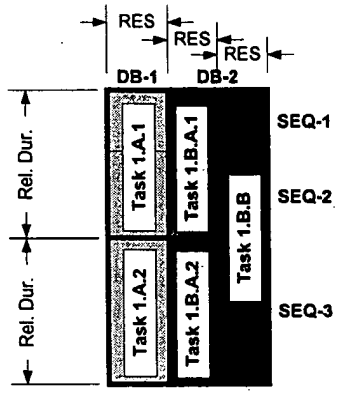
Processor 2 divides Task 1.B vertically (Box B); keeps Task 1.B.A

Task List	Result List
Task 1.A.2 Task 1.B.B	

Processor 1: Executing Task 1.A.1

Processor 2: Done dividing Task 1.B; about to divide Task 1.B.A

(e)



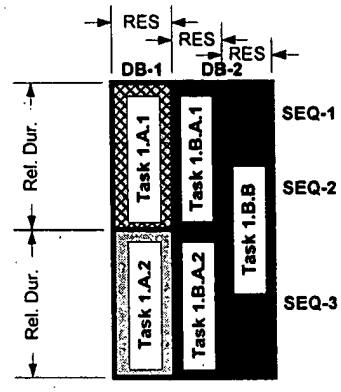
Processor 2 divides Task 1.B.A horizontally (Box C); keeps and begins executing Task 1.B.A.1

Task List	Result List
Task 1.A.2 Task 1.B.B Task 1.B.A.2	

Processor 1: Executing Task 1.A.1

Processor 2: Done dividing Task 1.B.A; about to execute Task 1.B.A.1

(f)



Processor 1 completes Task 1.A.1 (Box D); marks it READY (Box A)

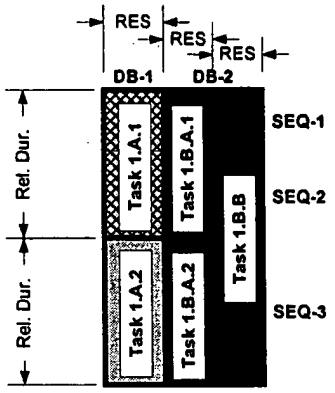
Task List	Result List
Task 1.A.2 Task 1.B.B Task 1.B.A.2	Task 1.A.1

Processor 1: Done executing Task 1.A.1; about to execute Task 1.B.A.2

Processor 2: Done dividing Task 1.B.A; about to execute Task 1.B.A.1

Figure 3 (Part 4 of 7)

(g)



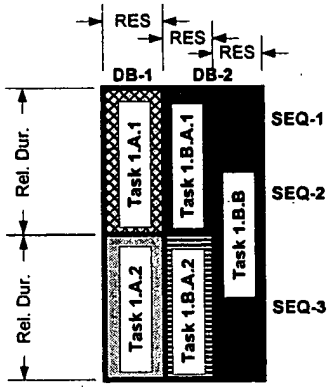
Processor 2 completes Task 1.B.A.1 (Box D); marks it READY (Box A)

Task List	Result List
Task 1.A.2 Task 1.B.B	Task 1.A.1 Task 1.B.A.1

Processor 1: Executing Task 1.B.A.2

Processor 2: Done executing Task 1.B.A.1; about to execute Task 1.A.2

(h)



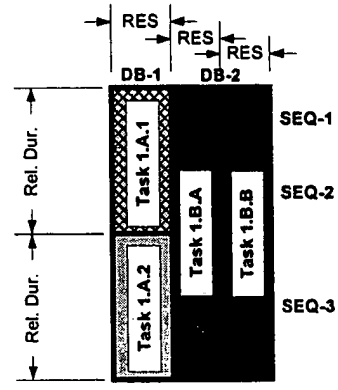
Processor 1 completes Task 1.B.A.2 (Box D)

Task List	Result List
Task 1.B.B	Task 1.A.1 Task 1.B.A.1

Processor 1: Done executing Task 1.B.A.2; about to merge with result of Task 1.B.A.1

Processor 2: Executing Task 1.A.2

(i)



Processor 1 merges result for Task 1.B.A.2 with result for Buddy Task 1.B.A.1, th reby computing result for Parent Task 1.B.A (Box E); marks Task 1.B.A READY since Buddy Task 1.B.B is not READY

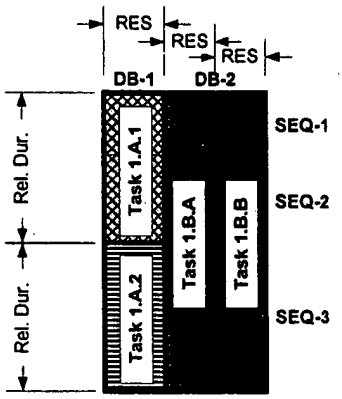
Task List	Result List
Task 1.B.B	Task 1.A.1 Task 1.B.A

Processor 1: Done forming result for Task 1.B.A; about to divide Task 1.B.B

Processor 2: Executing Task 1.A.2

Figure 3 (Part 5 of 7)

(j)



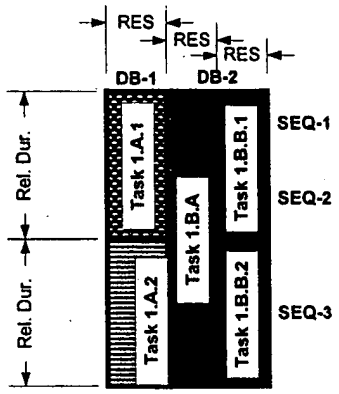
Processor 2 completes Task 1.A.2 (Box D)

Task List	Result List
	Task 1.A.1 Task 1.B.A

Processor 1: Dividing Task 1.B.B

Processor 2: Done executing Task 1.A.2; about to merge with result of Task 1.A.1

(k)



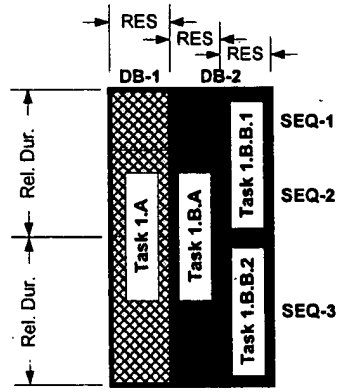
Processor 1 divides Task 1.B.B horizontally (Box C); keeps Task 1.B.B.1

Task List	Result List
Task 1.B.B.2	Task 1.B.A

Processor 1: Done dividing Task 1.B.B; about to execute Task 1.B.B.1

Processor 2: Merging results of Task 1.A.1 and Task 1.A.2

(l)



Proc ssor 2 merges result for Task 1.A.2 with result for Buddy Task 1.A.1, th reby computing result for Parent Task 1.A (Box E); marks Task 1.A READY, since its Buddy Task 1.B is not READY.

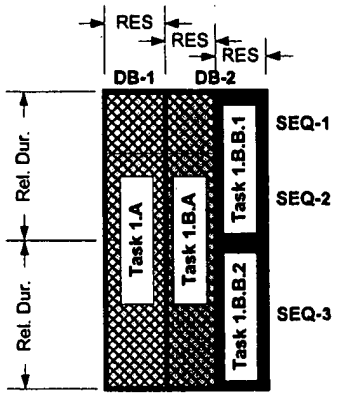
Task List	Result List
Task 1.B.B.2	Task 1.B.A Task 1.A

Processor 1: Executing Task 1.B.B.1

Processor 2: Done forming result for Task 1.A; about to execute Task 1.B.B.2

Figure 3 (Part 6 of 7)

(m)



Processor 1 completes Task 1.B.B.1 (Box D); marks it READY (Box A) since Task 1.B.B.2 is not READY

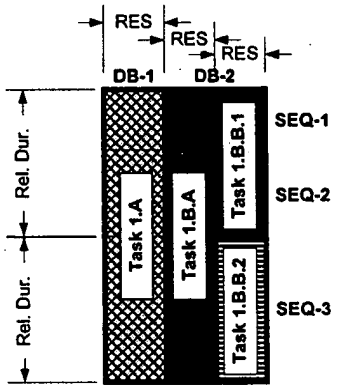
Task List

Result List
Task 1.B.A
Task 1.A
Task 1.B.B.1

Processor 1: Done executing Task 1.B.B.1; about to become idle

Processor 2: Executing Task 1.B.B.2

(n)



Processor 2 completes Task 1.B.B.2 (Box D)

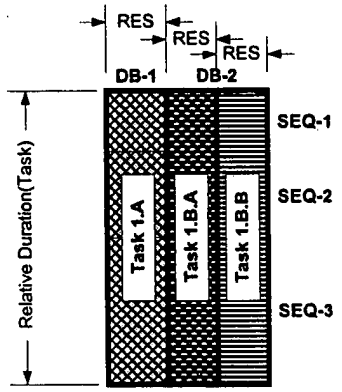
Task List

Result List
Task 1.B.A
Task 1.A
Task 1.B.B.1

Processor 1: Idle

Processor 2: Done executing Task 1.B.B.2; about to merge with result for Task 1.B.B.1

(o)



Processor 2 merges result for Task 1.B.B.1 with result for Buddy Task 1.B.B.2, thereby computing result for Parent Task 1.B.B (Box E)

Task List

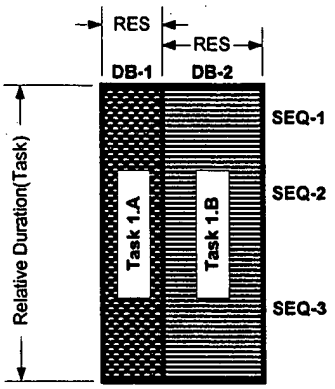
Result List
Task 1.B.A
Task 1.A

Processor 1: Idle

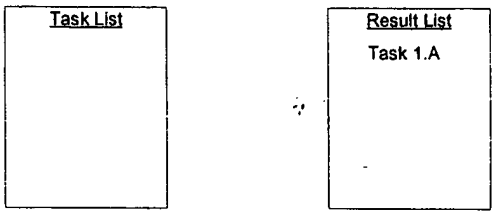
Processor 2: Done merging results for Task 1.B.B.1 and Task 1.B.B.2; about to merge with Task 1.B.A

Figure 3 (Part 7 of 7)

(p)



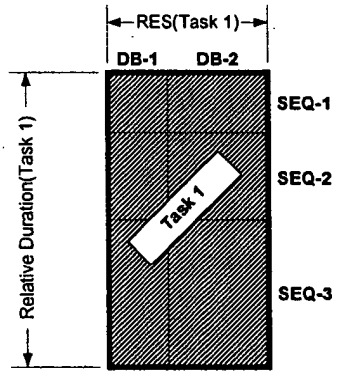
Since Task 1.B.B's Buddy Task 1.B.A is READY, Processor 2 merges result for Task 1.B.B with result for Buddy Task 1.B.A, thereby computing result for Parent Task 1.B (Box E)



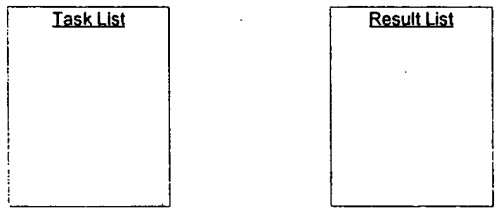
Processor 1: Idle

Processor 2: Done merging results for Task 1.B.B and Task 1.B.A; about to merge with Task 1.A

(q)



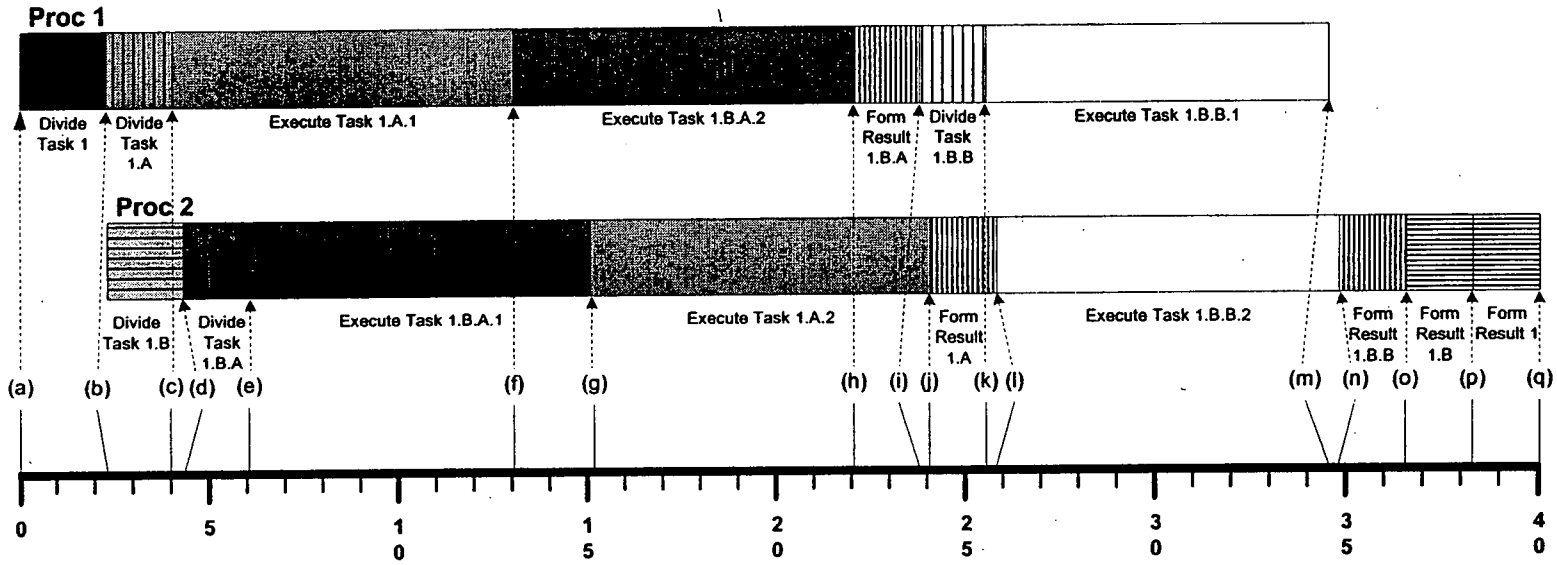
Since Task 1.B's Buddy Task 1.A is READY, Processor 2 merges result for Task 1.B with result for Buddy Task 1.A, thereby computing result for Parent Task 1 (Box E). This completes the computation, since Task 1 is the Entire Task and has no Buddy Task.



Processor 1: Idle

Processor 2: Done with final merging; now idle

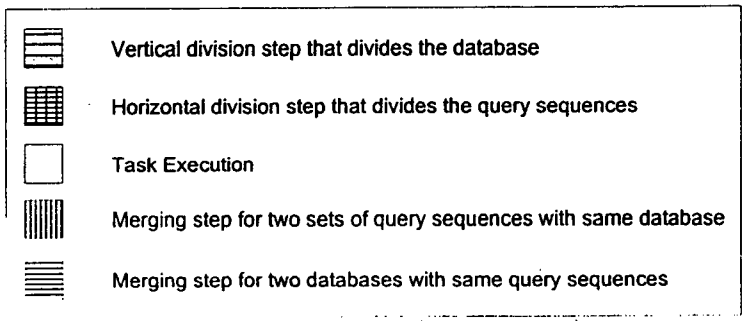
Figure 4
Processor Activity During Example Execution of Method



Time Scale for Computation (Letters refer to subfigures of Figure 2)

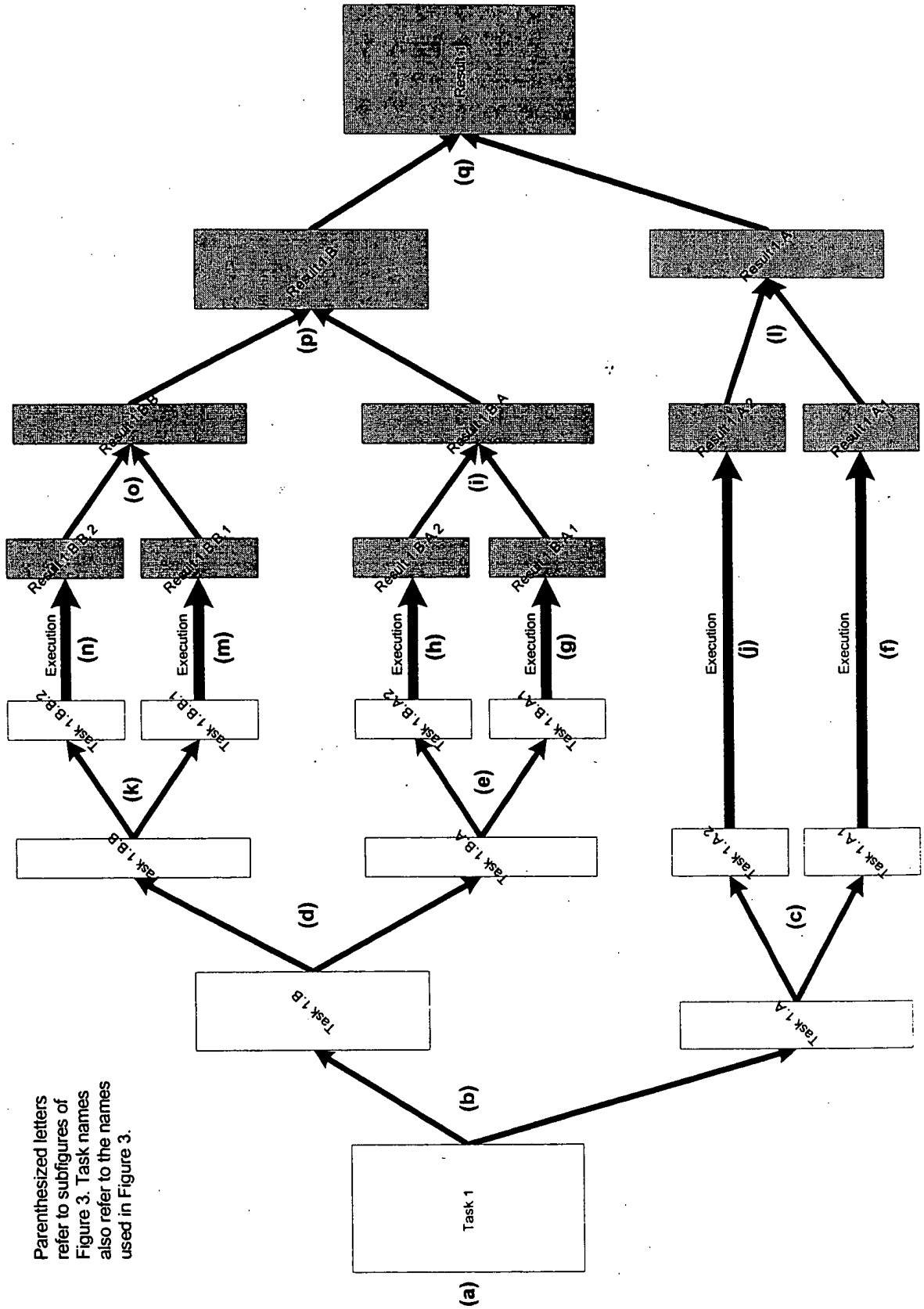
This figure contains timelines that illustrate the activities carried out on each of two processors during application of the method of the invention to compute the result of the entire searching task as illustrated in Figure 3. The markings for each activity are described below. In this figure, the fill pattern for each activity reflects the type of activity, while the fill coloring is used to indicate related activities. (Neither the fill patterns nor the fill colorings match those in Figure 3.)

The time scale does not represent actual time, but is intended to portray possible relative times at which various activities might take place. The time scale is consistent with the details of Figure 3 and with a possible operation of actual computer software implementing the method. The timelines are correlated with Figure 3 according to the parenthesized letters (e.g., "(a)") that correspond to the similarly designated subfigures of Figure 3.



Attorney Docket No.: 053706-5001

Figure 5
Representation of Task Division and Result Merging for Example of Figure 3



Parenthesized letters refer to subfigures of Figure 3. Task names also refer to the names used in Figure 3.

Figure 6(a): Plot of Times for Benchmark Example

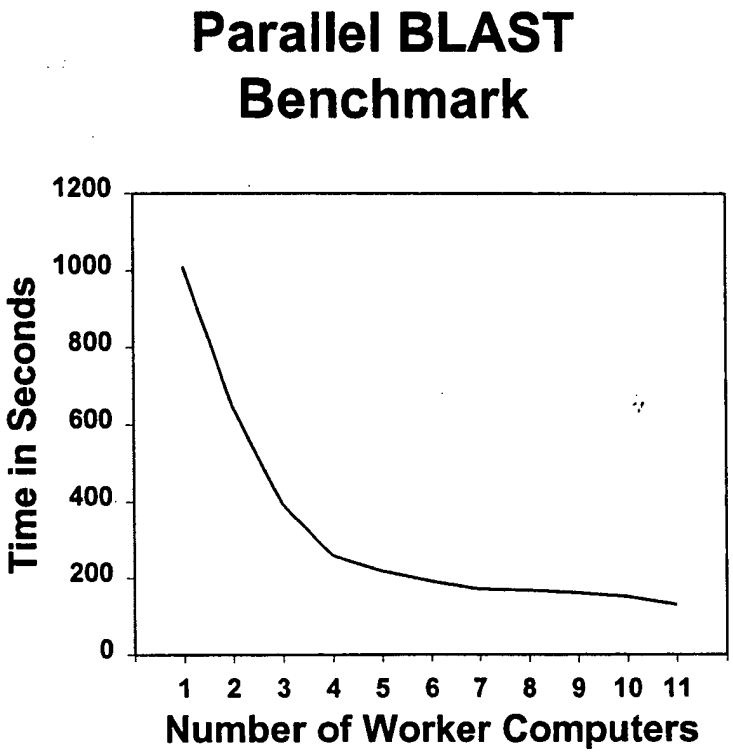
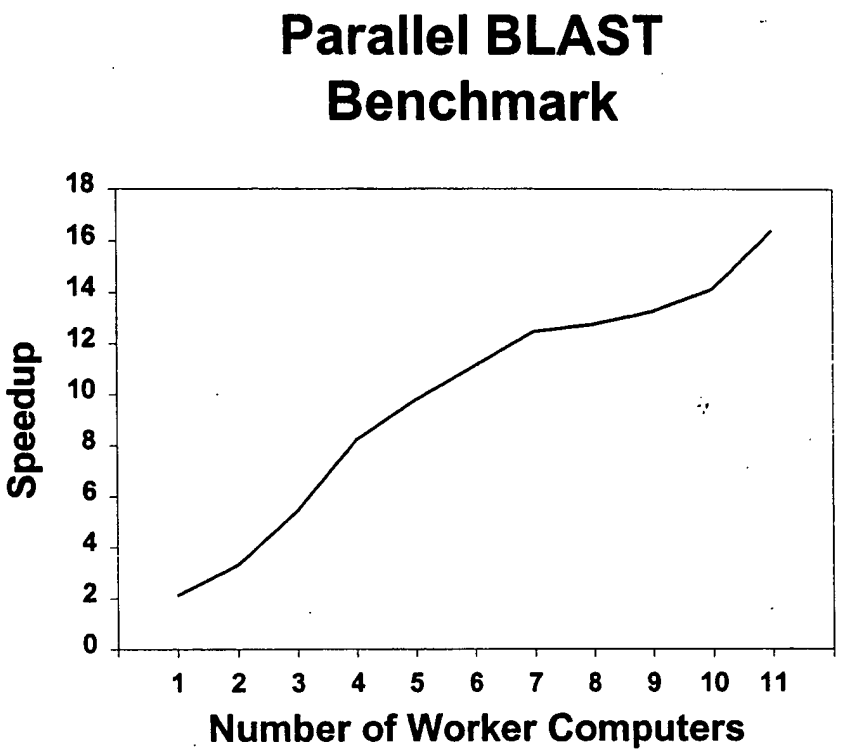



Figure 6(b): Plot of Speedup Values for Benchmark Example